***CITRUS FRAMEWORK***

* Protocols: HTTP-REST, SOAP, JMS, SMTP, FTP
* Helps to Integrate over any messaging protocol and format of data.

**Feature:**

* We can combine all protocols test, ex: if get rest call and we can pass that response through FTP
* It’s Java and spring framework based. It’s a spring framework.
* When we create a project, it’s basically a spring project.
* We can write test cases in **XML or Java**
* If a protocol is not supported by Citrus which you are working on, we can write own reusable component to communicate with protocol.

**Project Creation:**

* We need Maven for setup, Setup Maven with Eclipse
* Get Command from citrus website, (mvn archetype:generate -Dfilter**=**com.consol.citrus.archetypes:citrus)
* Create a Workspace(folder) in your machine, and navigate to that in CMD
* Then execute above command, and select proper option for project type and version.
* Give Artefact ID(Project) and Group ID(Package)
* Then Click Yes, Yes
* Once see “Build Success”
* Go to Eclipse, and import this project as existing maven project.

**Basics of Citrus:**

* Same like println, in citrus we use **actions echo** to print something in console
* We use **Spring Expression** Language to **${now}** for parameterization
* We can develop test cases in Java
* When we develop in Java we have to run them by **XML** **class** but providing test case names.
  + @CitrusXmlTest(name = "SampleXmlIT")
* Citrus uses **TestNG**/**Junit** as Testing Framework, so we can utilize more **annotations** in Test cases development
* When we use TestNG it inherit **“TestNGCitrusSupport** “class
* When we use Junit it inherit **“Junit4CitrusSupport”** class
* If we need to specify any Method as citrus test have to use **“@CitrusTest”**
* **Echo** to print in Java
  + *echo*("Today is: ${now}")

**TestNG/Junit Citrus Support:**

* Provides **“Builder Pattern Method”** to build test cases
* It builds complete test cases in design time, means it executes all actions first then it execute test cases
* Collect all test actions first in internal memory cache, then executes whole test case.

**Creation of Variable:**

* To create a variable, have to use **“Variable”** function
  + variable("now", "citrus:currentDate()");

**Runner Designer:**

* Solves the issues that comes with the test designer
* Executes each test action immediately
* To mix java DSL method calls and normal java code statement in our test in an unlimited way
* No wrapping within a custom action